# Pattern Recognition and Machine Learning Assignment

Ioannis Michalainas, Maria Charisi

December 2024

# PART A

In this part, our study involves the quantification of levels of stress of video game players. Our task in is to implement a Maximum Likelihood Estimator to diversify between two classes $\omega_1$ (stress) and $\omega_2$ (no stress). We need to be able to accurately classify a player based on his play-style as either stressed or not stressed (binary classification).

# Data

To perform our analysis we use the following data

- The Probability Density Function (PDF given $\theta$)

$$p(x \mid \theta) = \frac{\pi}{1 + (x - \theta)^2}$$

- Training Sets $D_1$ and $D_2$ for both classes ($\omega_1$, $\omega_2$)
- A Discriminant Function

$$g(x) = \log P(x \mid \hat{\theta}_1) - \log P(x \mid \hat{\theta}_2) + \log P(\omega_1) - \log P(\omega_2)$$

# A1

In this part we want to estimate variables $\theta_1$ and $\theta_2$. We begin by implementing the log likelihood function:

$$\log L(\theta \mid D) = \sum_{x \in D} \log p(x \mid \theta)$$

We now need to find $\theta_1$ and $\theta_2$ that maximize this function. A first approach would be to use the gradient of $logL(\theta \mid D)$ to find the optimal $\theta$ values. However, the gradient of this function does not have a closed-form expression, making it computationally challenging to apply this method directly.

# $logL(\theta \mid D)$ maximization

Instead, we take a simpler approach:

- Define a range of candidate $\theta$ values that likely contains the true $\theta$.
- Evaluate the log-likelihood function for these candidates and select the $\theta$ that maximizes it.

Since the data range spans approximately [-4.5, 4.1], we select a slightly wider candidate range for $\theta$, such as [-5, 5], to ensure it includes the optimal value.

# Plot

$\hat{\theta}_1$ estimation: 2.60 $\hat{\theta}_2$ estimation: -3.16

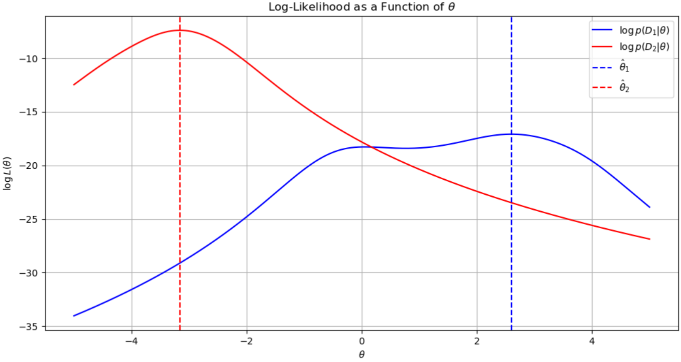


Figure: Visualizing Optimal $\theta$ Values

# A2

Now, we need to use Discriminant Function

$$g(x) = \log P(x \mid \hat{\theta}_1) - \log P(x \mid \hat{\theta}_2) + \log P(\omega_1) - \log P(\omega_2)$$

to classify our data.

# Priors Calculation

We calculate the *a priori probabilities* for each class $\omega_i$. We have a total of 12 samples:

- 7 classified in $\omega_1$
- 5 classified in $\omega_2$

So we calculate $P(\omega_1) = 7/12$ and $P(\omega_2) = 5/12$

# Plot

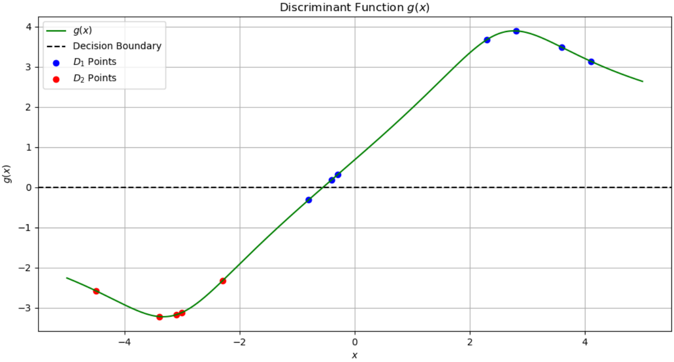


Figure: Discriminant Function Visualization

# Observations

- Most of the $D1$ points are classified correctly, as $g(x) > 0$.
- All of the $D2$ points are classified correctly, as $g(x) < 0$.

**Decision Rule:** The decision boundary is at $g(x) = 0$. For any $x$:

- If $g(x) > 0$, classify $x$ as 1 (no stress).
- If $g(x) < 0$, classify $x$ as 2 (stress).

# Conclusion

The classification rule leads to some misclassifications. While the decision rule works well for most of the data (11/12, 92%), there are always some trade-offs in classification accuracy. Achieving perfect classification is sometimes not feasible or desirable.

- Attempting to perfectly classify all points might lead to **overfitting**. A model that fits all training data perfectly may not generalize well to unseen data.
- The data may inherently contain some ambiguous or **overlapping** cases that no model can perfectly classify, especially if the two classes are not linearly separable.

# Part B

In this part our task is to implement a new classifier for the previous problem, this time using Bayesian Estimation to estimate parameter $\theta$.

# Data

To perform our analysis we use the following data:

- The Probability Density Functions (PDF)

$$p(\theta) = \frac{1}{10\pi\left(1 + \left(\frac{\theta}{10}\right)^2\right)}, \quad p(x \mid \theta) = \frac{\pi}{1 + (x - \theta)^2}$$

- Training Samples $D_1$ and $D_2$ for both classes $(\omega_1, \omega_2)$
- A Discriminant Function

$$h(x) = \log P(x \mid D_1) - \log P(x \mid D_2) + \log P(\omega_1) - \log P(\omega_2)$$

# B1

In this part, our goal is to calculate the a posteriori probability of $\theta$,

$$p(\theta \mid D) = \frac{p(D \mid \theta) \cdot p(\theta)}{\int_{-\infty}^{\infty} p(D \mid \theta) \cdot p(\theta) \, d\theta}$$

# $\theta$ Candidates

- In Part A, the dataset values were relatively small ([-4.5, 4.1]). Using a range of [-5, 5] was a practical choice because it encompassed the likely $\theta$ values where the likelihood peaks.
- In Part B, the prior distribution has a broader support (($\theta/10$) in the denominator suggests a wider possible range). To ensure the posterior distribution adequately integrates both the likelihood and the prior, we expanded the range to [-10, 10].

# Plot

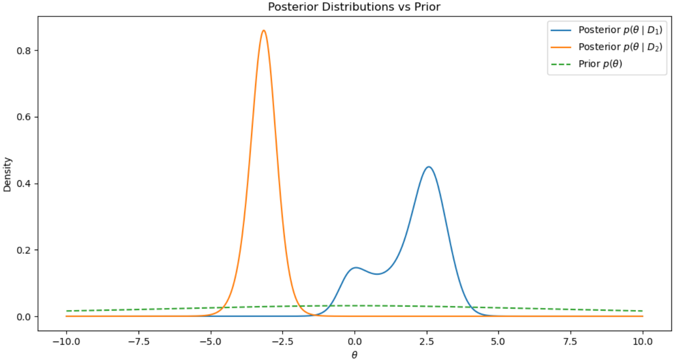


Figure: $P(\theta \mid D_1)$, $P(\theta \mid D_2)$ and $P(\theta)$

# Observations

- The prior distribution $p(\theta)$ is flat and spread out over the range of $\theta$ values. It shows minimal preference for any specific $\theta$, reflecting the prior belief before observing the data.
- The posteriors are much more concentrated than the prior, reflecting how the observed data updates the prior belief and provides more precise estimates of $p(\theta)$.
- The location of the peaks in the posteriors ($p(\theta \mid D_1)$ near 2 and $p(\theta \mid D_2)$ near -3) shows the influence of the datasets $D_1$ and $D_2$, respectively.

# B2

We declare posterior predictive distribution $p(x \mid D)$ like so:

$$p(x \mid D) = \int p(x \mid \theta) p(\theta \mid D) d\theta$$

Then we declare the discriminant function $h(x)$:

$$h(x) = \log P(x \mid D_1) - \log P(x \mid D_2) + \log P(\omega_1) - \log P(\omega_2)$$

# Plot

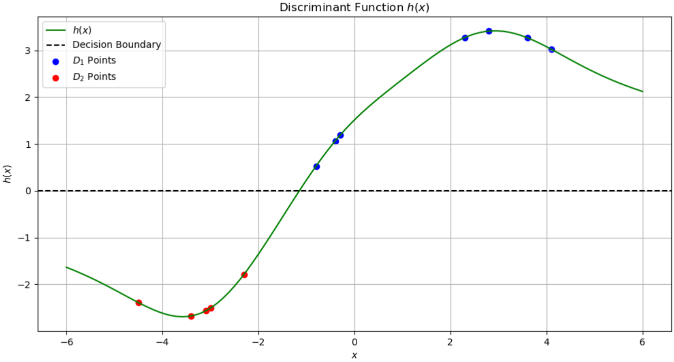


Figure: Discriminant Function Visualization

# Observations

- All of the $D1$ points are classified correctly, as $h(x) > 0$.
- All of the $D2$ points are classified correctly, as $h(x) < 0$.

**Decision Rule:** The decision boundary is at $h(x) = 0$. For any $x$:

- If $h(x) > 0$, classify $x$ as 1 (no stress).
- If $h(x) < 0$, classify $x$ as 2 (stress).

# Maximum Likelihood Estimation vs Bayesian Estimation

**Part A:**

- Estimates a single value for $\theta$ ($\hat{\theta}_1$ and $\hat{\theta}_2$) that maximizes the likelihood for each class.
- The decision boundary is determined by the log-likelihood and prior probabilities.

**Part B:**

- Considers the entire distribution of $\theta$ given the data (the posterior distribution) to make predictions.
- The decision boundary is determined by the posterior predictive distribution, which integrates over all possible values of $\theta$ weighted by their posterior probabilities.

We attribute the better performance of BE to the fact that we have prior knowledge about the parameter $\theta$, in means of $p(\theta)$.

# Part C

In this part, our study focuses on the classification of three Iris species: Iris setosa, Iris versicolor, and Iris virginica (i.e. three *classes*).

- Using the **sklearn** library, we analyze a dataset of 150 measurements (50 per species).
- Only the first two features (sepal length and width) are used.
- A DecisionTreeClassifier is implemented to classify 50% of randomly selected samples after training the model on the remaining 50%.

# C1.1

Our goal is to find the optimal depth with respect to *accuracy*. To do so, we iterate in a range of possible depths (1,10), where the optimal depth should lie. We chose this particular range to avoid overfitting.

- Deep descision trees generally lead to overfitting -they tend to capture overly specific patterns that do not generalize well.
- This is especially true in our case, where the dataset is relatively small (150 samples).

# Plot

We create the **Decision Tree** that yields the best *accuracy*. As shown in the plot, the optimal **depth** is 3 with **accurracy** 0.79.

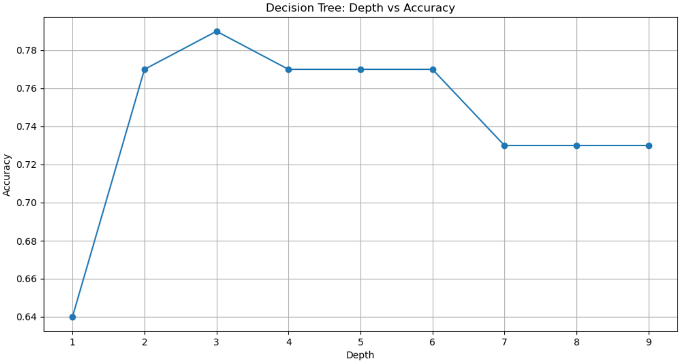


Figure: Optimal Depth

# C1.2

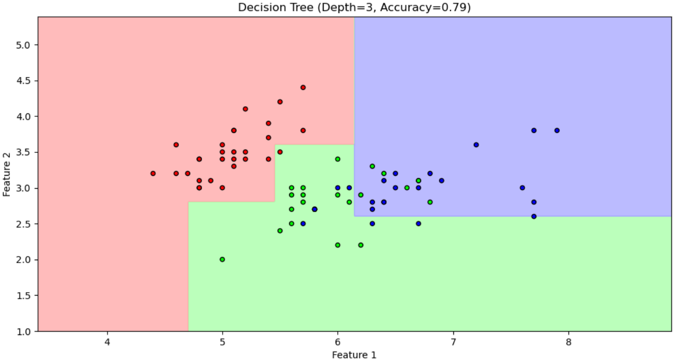


Figure: Classification

# C2.1

In this task, we extend our classification of the three Iris species using a **Random Forest** classifier with *100 decision trees*.

- From the training set used previously (set A, 50% of the dataset), we create 100 training sets, each consisting of $\gamma = 50\%$ of set A, using the **bootstrap** method.
- Each decision tree is trained on its respective bootstrap set with a fixed maximum depth.
- The test set from the previous task is reused to evaluate the performance of the Random Forest classifier.

# Optimal Depth

Our goal is yet again to find the optimal depth with respect to *accuracy*. To do so, we iterate in a range of possible depths (1,10), where the optimal depth should lie.

# Plot

We create the **Random Forest** that yields the best *accuracy*. As shown in the plot, the optimal **depth** is 2 with **accurracy** 0.83.

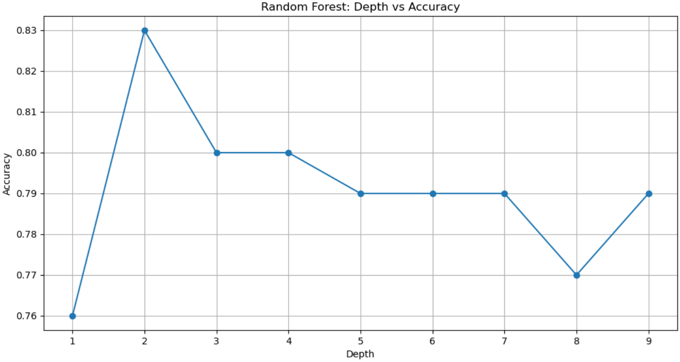


Figure: Optimal Depth

# C2.2

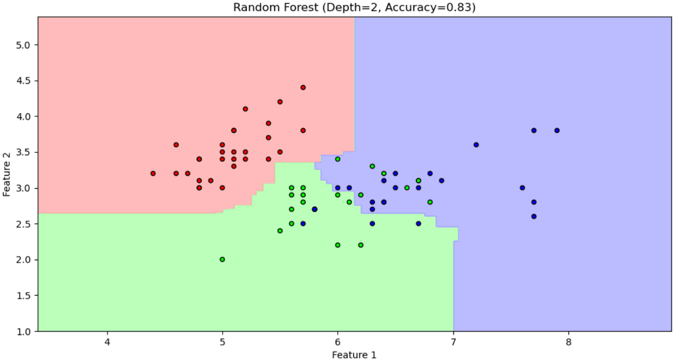


Figure: Enter Caption

# C2.3

We calculate the *accuracy* achieved for different values of $\gamma$ on its respective best depth. Essentially, we keep the *best* result we can get for each $\gamma$, as an example.

- For lower values of $\gamma$ (0.1, 0.2, 0.3), the best accuracy varies between 0.80 and 0.81, with slight improvements as gamma increases.
- For higher values (0.4 and above), the accuracy stabilizes at around 0.83, indicating diminishing returns in performance improvement with increasing gamma.
- The best depth remains practically constant and equal to 2 for almost every $\gamma$.

# $\gamma$ Influence

- Generally, small $\gamma$ leads to **higher bias**, because each bootstrap set contains less information about the entire dataset.
- On the other hand, for bigger values of $\gamma$ the diversity among bootstrap sets decreases, which can reduce the ensemble's effectiveness at reducing variance.

In conclusion, increasing $\gamma$ up to 0.4 seems to benefit our algorithm, but further increasing it does not have any effect on *accuracy* or the best depth for the Random Forest.

# PART D

This is the introduction slide.

# 2

# 3

# 4

# 5

# 6

# 7

# 8

# 9

# 10

# 11

# 12

# 13

# 14

# 15

# 16

# 17

# 18

# 19

# 20